Package 'memoria'

October 13, 2022

Type Package

Title Quantifying Ecological Memory in Palaeoecological Datasets and Other Long Time-Series

Version 1.0.0

Author Blas M. Benito

Maintainer Blas M. Benito <blasbenito@gmail.com>

Description Tools to quantify ecological memory in long time-series with Random Forest models (Breiman 2001 <doi:10.1023/A:1010933404324>) fitted with the 'ranger' library (Wright and Ziegler 2017 <doi:10.18637/jss.v077.i01>). Particularly oriented to palaeoecological datasets and simulated pollen curves produced by the 'virtualPollen' package, but also applicable to other long time-series involving a set of environmental drivers and a biotic response.

License GPL (>= 2)

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

VignetteBuilder knitr

Depends R (>= 2.10)

Imports ggplot2, ranger, cowplot, viridis, viridisLite, zoo, stringr, HH, tidyr

Suggests devtools, formatR, kableExtra, magrittr, knitr, rmarkdown, rpart, rpart.plot, randomForest, virtualPollen

NeedsCompilation no

Repository CRAN

Date/Publication 2019-05-17 08:00:02 UTC

R topics documented:

climate 2
computeMemory 3
experimentToTable 5

extractMemoryFeatures 6
mergePalaeoData 9
palaeodata 10
palaeodataLagged 11
palaeodataMemory 12
plotExperiment 13
plotInteraction 14
plotMemory 15
pollen 16
prepareLaggedData 17
runExperiment 18

Index **21**

`climate` *Dataframe with palaeoclimatic data.*

Description

A dataframe containing palaeoclimate data at 1 ky temporal resolution with the following columns:

Usage

```
data(climate)
```

Format

dataframe with 6 columns and 800 rows.

Details

- *age* in kiloyears before present (ky BP).
- *temperatureAverage* average annual temperature in Celsius degrees.
- *rainfallAverage* average annual precipitation in milimetres per day (mm/day).
- *temperatureWarmestMonth* average temperature of the warmest month, in Celsius degrees.
- *temperatureColdestMonth* average temperature of the coldest month, in Celsius degrees.
- *oxigenIsotope* delta O18, global ratio of stable isotopes in the sea floor, see `http://lorraine-lisiecki.com/stack.html` for further details.

Author(s)

Blas M. Benito <blasbenito@gmail.com>

computeMemory *Quantifies ecological memory with Random Forest.*

Description

Takes the oputput of prepareLaggedData to fit the following model with Random Forest:

$p_t = p_{t-1} + ... + p_{t-n} + d_t + d_{t-1} + ... + d_{t-n} + r$

where:

- d is a driver (several drivers can be added).
- t is the time of any given value of the response p.
- $t - 1$ is the lag number 1 (in time units).
- $p_{t-1} + ... + p_{t-n}$ represents the endogenous component of ecological memory.
- $d_{t-1} + ... + d_{t-n}$ represents the exogenous component of ecological memory.
- d_t represents the concurrent effect of the driver over the response.
- r represents a column of random values, used to test the significance of the variable importance scores returned by Random Forest.

Usage

```
computeMemory(
 lagged.data = NULL,
 drivers = NULL,
 response = "Response",
 add.random = TRUE,
 random.mode = "autocorrelated",
 repetitions = 10,
 subset.response = "none",
 min.node.size = 5,
 num.trees = 2000,
 mtry = 2
 )
```

Arguments

lagged.data	a lagged dataset resulting from prepareLaggedData. See palaeodataLagged as example.
drivers	a character string or vector of character strings with variables to be used as predictors in the model (i.e. c("Suitability", "Driver.A")). **Important:** drivers names must not have the character "_".
response	character string, name of the response variable (typically, "Response_0").
add.random	if TRUE, adds a random term to the model, useful to assess the significance of the variable importance scores.

random.mode	either "white.noise" or "autocorrelated". See details.
repetitions	integer, number of random forest models to fit.
subset.response	character string with values "up", "down" or "none", triggers the subsetting of the input dataset. "up" only models memory on cases where the response's trend is positive, "down" selectes cases with negative trends, and "none" selects all cases.
min.node.size	integer, argument of the ranger function. Minimal number of samples to be allocated in a terminal node. Default is 5.
num.trees	integer, argument of the ranger function. Number of regression trees to be fitted (size of the forest). Default is 2000.
mtry	integer, argument of the ranger function. Number of variables to possibly split at in each node. Default is 2.

Details

This function uses the ranger package to fit Random Forest models. Please, check the help of the ranger function to better understand how Random Forest is parameterized in this library. This function fits the model explained above as many times as defined in the argument `repetitions`. To test the statistical significance of the variable importance scores returned by random forest, on each repetition the model is fitted with a different `r` (random) term. If `random.mode` equals "autocorrelated", the random term will have a temporal autocorrelation, and if it equals "white.noise", it will be a pseudo-random sequence of numbers generated with `rnorm`, with no temporal autocorrelation. The importance of the random sequence (as computed by random forest) is stored for each model run, and used as a benchmark to assess the importance of the other predictors used in the models. Importance values of other predictors that are above the median of the importance of the random term should be interpreted as non-random, and therefore, significant.

Value

A list with 4 slots:

- `memory` dataframe with five columns:
  - `Variable` character, names and lags of the different variables used to model ecological memory.
  - `median` numeric, median importance across `repetitions` of the given `Variable` according to Random Forest.
  - `sd` numeric, standard deviation of the importance values of the given `Variable` across `repetitions`.
  - `min` and `max` numeric, percentiles 0.05 and 0.95 of importance values of the given `Variable` across `repetitions`.
- `R2` vector, values of pseudo R-squared value obtained for the Random Forest model fitted on each repetition. Pseudo R-squared is the Pearson correlation beteween the observed and predicted data.
- `prediction` dataframe, with the same columns as the dataframe in the slot `memory`, with the median and confidence intervals of the predictions of all random forest models fitted.
- `multicollinearity` multicollinearity analysis on the input data performed with vif. A vif value higher than 5 indicates that the given variable is highly correlated with other variables.

Author(s)

Blas M. Benito <blasbenito@gmail.com>

See Also

plotMemory, extractMemoryFeatures

##'

- Wright, M. N. & Ziegler, A. (2017). ranger: A fast implementation of random forests for high dimensional data in C++ and R. J Stat Softw 77:1-17. https://doi.org/10.18637/jss.v077.i01.
- Breiman, L. (2001). Random forests. Mach Learn, 45:5-32. https://doi.org/10.1023/A:1010933404324.
- Hastie, T., Tibshirani, R., Friedman, J. (2009). The Elements of Statistical Learning. Springer, New York. 2nd edition.

Examples

```
#loading data
data(palaeodataLagged)

memory.output <- computeMemory(
 lagged.data = palaeodataLagged,
 drivers = c("climate.temperatureAverage", "climate.rainfallAverage"),
 response = "Response",
 add.random = TRUE,
 random.mode = "autocorrelated",
 repetitions = 10,
 subset.response = "none"
)

str(memory.output)
str(memory.output$memory)

#plotting output
plotMemory(memory.output = memory.output)
```

---

experimentToTable *Turns the outcome of* runExperiment *into a long table.*

Description

Takes the output of runExperiment, extracts the dataframes containing the ecological memory patterns generated by computeMemory, and binds them together into a single dataframe ready for further analyses or plotting.

Usage

```
experimentToTable(
 experiment.output = NULL,
 parameters.file = NULL,
 sampling.names = NULL,
 R2 = TRUE
 )
```

Arguments

`experiment.output`	list, output of `runExperiment`.
`parameters.file`	dataframe of simulation parameters.
`sampling.names`	vector of character strings with the names of the columns of `simulations.file`.
`R2`	boolean. If TRUE, the average pseudo R-squared of the random forest models used to analyze the ecological memory pattern of the virtual taxa are shown with the taxon traits.

Details

This function is used internally by `plotExperiment`, but it is also available to users in case they want to do other kinds of analyses or plots with the data.

Value

A dataframe.

Author(s)

Blas M. Benito <blasbenito@gmail.com>

See Also

`runExperiment`, `plotExperiment`

`extractMemoryFeatures` *Extracts ecological memory features on the output of* `computeMemory`.

Description

It computes the following features of the ecological memory patterns returned by `computeMemory`:

- `memory strength` maximum difference in relative importance between each component (endogenous, exogenous, and concurrent) and the median of the random component. This is computed for exogenous, endogenous, and concurrent effect.

- memory length proportion of lags over which the importance of a memory component is above the median of the random component. This is only computed for endogenous and exogenous memory.
- dominance proportion of the lags above the median of the random term over which a memory component has a higher importance than the other component. This is only computed for endogenous and exogenous memory.

Usage

```
extractMemoryFeatures(
 memory.pattern = NULL,
 exogenous.component = NULL,
 endogenous.component = NULL,
 sampling.subset = NULL,
 scale.strength = TRUE
 )
```

Arguments

memory.pattern	either a list resulting from computeMemory, or a dataframe with memory patterns of several taxa generated by experimentToTable.
exogenous.component	character string or vector of character strings, name of the variable or variables defining the exogenous component.
endogenous.component	character string, string, name of the variable defining the endogenous component. If the data was generated by prepareLaggedData, endogenous.component would usually be "Response".
sampling.subset	only relevant when analysis.output is the result of runExperiment. Character string with the name of the column of the list with the simulation outcomes.
scale.strength	boolean. If TRUE, the strength of the ecological memory components, which has the same units as the importance scores yielded by Random Forest (percentage of increment in mean squared error when a variable is permuted), is scaled between 0 and 1.

Details

Warning: this function only works when only one exogenous component (driver) is used to define the model in computeMemory. If more than one driver is provided throught the argument exogenous.component, the maximum importance scores of all exogenous variables is considered. In other words, the importance of exogenous variables is not additive.

Value

A dataframe with 8 columns and 1 row if memory.pattern is the output of computeMemory and 13 columns and as many rows as taxa are in the input if it is the output of experimentToTable. The columns are:

- *label* character string to identify the taxon. It either inherits its values from `experimentToTable`, or sets the default ID as "1".
- *strength.endogenous* numeric in the range [0, 100], in importance units (percentage of increment in the mean squared error of the random forest model if the variable is permuted) difference between the maximum importance of the endogenous component at any lag and the median of the random component (see details in `computeMemory`)
- *strength.exogenous* numeric in the range [0, 100], same as above, but for the exogenous component.
- *strenght.concurrent* numeric in the range [0, 100], same as above, but for the concurrent component (driver at lag 0).
- *length.endogenous* numeric in the range [0, 100], proportion of lags over which the importance of the endogenous memory component is above the median of the random component.
- *length.exogenous* numeric in the range [0, 1], same as above but for the exogenous memory component.
- *dominance.endogenous* numeric in the range [0, 1], proportion of the lags above the median of the random term over which a the endogenous memory component has a higher importance than the exogenous component.
- *dominance.exogenous*, opposite as above.
- *maximum.age*, numeric. As every column after this one, only provided if `memory.pattern` is the output of `experimentToTable`. Trait of the given taxon.
- *fecundity* numeric, trait of the given taxon.
- *niche.A.mean* numeric, trait of the given taxon.
- *niche.A.sd* numeric, trait of the given taxon.
- *sampling* numeric, trait of the given taxon.

Author(s)

Blas M. Benito <blasbenito@gmail.com>

See Also

`computeMemory`

Examples

```
#loading example data
data(palaeodataMemory)

#computing ecological memory features
memory.features <- extractMemoryFeatures(
  memory.pattern = palaeodataMemory,
  exogenous.component = c(
  "climate.temperatureAverage",
  "climate.rainfallAverage"
  ),
```

```
endogenous.component = "Response",
sampling.subset = NULL,
scale.strength = TRUE
)
```

`mergePalaeoData` *Merges palaeoecological datasets with different time resolution.*

Description

It merges palaeoecological datasets with different time intervals between consecutive samples into a single dataset with samples separated by regular time intervals defined by the user

Usage

```
mergePalaeoData(
 datasets.list = NULL,
 time.column = NULL,
 interpolation.interval = NULL
 )
```

Arguments

`datasets.list`	list of dataframes, as in `datasets.list = list(climate = climate.dataframe, pollen = pollen.dataframe)`. The provided dataframes must have an age/time column with the same column name and the same units of time. Non-numeric columns in these dataframes are ignored.
`time.column`	character string, name of the time/age column of the datasets provided in `datasets.list`.
`interpolation.interval`	temporal resolution of the output data, in the same units as the age/time columns of the input data

Details

This function fits a `loess` model of the form `y ~ x`, where `y` is any column given by `columns.to.interpolate` and `x` is the column given by the `time.column` argument. The model is used to interpolate column `y` on a regular time series of intervals equal to `interpolation.interval`. All columns in every provided dataset go through this process to generate the final data with samples separated by regular time intervals. Non-numeric columns are ignored, and absent from the output dataframe.

Value

A dataframe with every column of the initial dataset interpolated to a regular time grid of resolution defined by `interpolation.interval`. Column names follow the form datasetName.columnName, so the origin of columns can be tracked.

Author(s)

Blas M. Benito <blasbenito@gmail.com>

Examples

```
#loading data
data(pollen)
data(climate)

x <- mergePalaeoData(
 datasets.list = list(
   pollen=pollen,
   climate=climate
 ),
 time.column = "age",
 interpolation.interval = 0.2
 )
```

`palaeodata` *Dataframe with pollen and climate data.*

Description

A dataframe with a regular time grid of 0.2 ky resolution resulting from applying `mergePalaeoData` to the datasets `climate` and `pollen`:

Usage

```
data(palaeodata)
```

Format

dataframe with 10 columns and 7986 rows.

Details

- *age* in ky before present (ky BP).
- *pinus* pollen counts of Pinus.
- *quercus* pollen counts of Quercus.
- *poaceae* pollen counts of Poaceae.
- *artemisia* pollen counts of Artemisia.
- *temperatureAverage* average annual temperature in Celsius degrees.

- *rainfallAverage* average annual precipitation in milimetres per day (mm/day).
- *temperatureWarmestMonth* average temperature of the warmest month, in Celsius degrees.
- *temperatureColdestMonth* average temperature of the coldest month, in Celsius degrees.
- *oxigenIsotope* delta O18, global ratio of stable isotopes in the sea floor, see http://lorraine-lisiecki.com/stack.html for further details.

Author(s)

Blas M. Benito <blasbenito@gmail.com>

---

palaeodataLagged *Lagged data generated by* prepareLaggedData.

Description

A dataframe resulting from the application of prepareLaggedData to the dataset palaeodata. The dataframe columns are:

Usage

```
data(palaeodataLagged)
```

Format

dataframe with 13 columns and 3988 rows.

Details

- *Response_0* numeric, values of the response variable selected by the user in the argument response of the function prepareLaggedData. This column is used as response variable by the function computeMemory. In this case, Response represent pollen counts of Pinus.
- *Response_0.2-1* numeric, time delayed values of the response for different lags (in ky). Considered together these columns represent the endogenous ecological memory.
- *climate.temperatureAverage_0* numeric, values of the variable temperatureAverage for the lag 0 (no lag). This column represents the concurrent effect of the temperature over the response. #'
- *climate.rainfallAverage_0* numeric, values of the variable rainfallAverage for the lag 0 (no lag). This column represents the concurrent effect of rainfall over the response.
- *climate.temperatureAverage_0.2-1* numeric, time delayed values of temperatureAverage for lags 0.2 to 1 (in ky). #'
- *climate.rainfallAverage_0.2-1* numeric, time delayed values of rainfallAverage for lags 0.2 to 1 (in ky).

Author(s)

Blas M. Benito <blasbenito@gmail.com>

palaeodataMemory *Output of* computeMemory

Description

List containing the output of computeMemory applied to palaeodataLagged. Its slots are:

Usage

```
data(palaeodataMemory)
```

Format

List with four slots.

Details

- memory dataframe with five columns:
  - Variable character, names and lags of the different variables used to model ecological memory.
  - median numeric, median importance across repetitions of the given Variable according to Random Forest.
  - sd numeric, standard deviation of the importance values of the given Variable across repetitions.
  - min and max numeric, percentiles 0.05 and 0.95 of importance values of the given Variable across repetitions.
- R2 vector, values of pseudo R-squared value obtained for the Random Forest model fitted on each repetition. Pseudo R-squared is the Pearson correlation beteween the observed and predicted data.
- prediction dataframe, with the same columns as the dataframe in the slot memory, with the median and confidence intervals of the predictions of all random forest models fitted.
- multicollinearity multicollinearity analysis on the input data performed with vif. A vif value higher than 5 indicates that the given variable is highly correlated with other variables.

Author(s)

Blas M. Benito <blasbenito@gmail.com>

plotExperiment *Plots the output of* runExperiment.

Description

It takes the output of runExperiment, and generates plots of ecological memory patterns for a large number of simulated pollen curves.

Usage

```
plotExperiment(
 experiment.output = NULL,
 parameters.file = NULL,
 experiment.title = NULL,
 sampling.names = NULL,
 legend.position = "bottom",
 R2 = NULL,
 filename = NULL,
 strip.text.size = 12,
 axis.x.text.size = 8,
 axis.y.text.size = 12,
 axis.x.title.size = 14,
 axis.y.title.size = 14,
 title.size = 18,
 caption = ""
 )
```

Arguments

experiment.output	list, output of runExperiment.
parameters.file	dataframe of simulation parameters.
experiment.title	character string, title of the plot.
sampling.names	vector of character strings with the names of the columns used in the argument simulations.file of runExperiment. If NULL, no pdf plot is produced.
legend.position	legend position in ggplot object. One of "bottom", "right", "none".
R2	boolean. If TRUE, pseudo R-squared values are printed along with the traits of the virtual taxa.
filename	character string, path and name (without extension) of the output pdf file.
strip.text.size	size of the facet's labels.
axis.x.text.size	size of the labels in x axis.

`axis.y.text.size`
: size of the labels in y axis.

`axis.x.title.size`
: size of the title of the x axis.

`axis.y.title.size`
: size of the title of the y axis.

`title.size`
: size of the plot title.

`caption`
: character string, caption of the output figure.

Value

A ggplot2 object.

Author(s)

Blas M. Benito <blasbenito@gmail.com>

See Also

`plotMemory`, `runExperiment`

---

plotInteraction *Plots response surfaces for tree-based models.*

Description

Plots a response surface plot or interaction plot (2 predictors and a model response) for models of the functions `ranger`, `randomForest`, and `rpart`. It also plots the observed data on top of the predicted surface.

Usage

```
plotInteraction(
  model = NULL,
  data = NULL,
  x = NULL,
  y = NULL,
  z = NULL,
  grid = 100,
  point.size.range = c(0.1, 1)
  )
```

Arguments

model	a model object produced by the functions ranger, randomForest, or rpart.
data	dataframe used to fit the model.
x	character string, name of column in data to be plotted in the x axis.
y	character string, name of column in data to be plotted in the y axis.
z	character string, name of column in data to be plotted as a surface. Generally, the name of the response variable in model.
grid	numeric, resolution of the x and y axes.
point.size.range	numeric vector with two values defining the range size of the points representing the observed data.

Value

A ggplot object.

Author(s)

Blas M. Benito <blasbenito@gmail.com>

---

plotMemory *Plots output of* computeMemory

Description

Plots the ecological memory pattern yielded by computeMemory.

Usage

```
plotMemory(
  memory.output = NULL,
  title = "Ecological memory pattern",
  legend.position = "right",
  filename = NULL
)
```

Arguments

memory.output	a dataframe with one time series per column.
title	character string, name of the numeric column to be used as response in the model.
legend.position	character vector, names of the numeric columns to be used as predictors in the model.
filename	character string, name of output pdf file. If NULL or empty, no pdf is produced. It shouldn't include the extension of the output file.

Value

A ggplot object.

Author(s)

Blas M. Benito <blasbenito@gmail.com>

See Also

computeMemory

Examples

```
#loading data
data(palaeodataMemory)

#plotting memory pattern
plotMemory(memory.output = palaeodataMemory)
```

---

pollen *Dataframe with pollen counts.*

Description

A dataframe with the following columns:

Usage

```
data(pollen)
```

Format

dataframe with 5 columns and 639 rows.

Details

- *age* in kiloyears before present (ky BP).
- *pinus* pollen counts of Pinus.
- *quercus* pollen counts of Quercus.
- *poaceae* pollen counts of Poaceae.
- *artemisia* pollen counts of Artemisia.

Author(s)

Blas M. Benito <blasbenito@gmail.com>

prepareLaggedData *Organizes time series data into lags.*

Description

Takes a multivariate time series, where at least one variable is meant to be used as a response while the others are meant to be used as predictors in a model, and organizes it in time lags, generating one new column per lag and variable in the model.

Usage

```
prepareLaggedData(
  input.data = NULL,
  response = NULL,
  drivers = NULL,
  time = NULL,
  oldest.sample = "first",
  lags = NULL,
  time.zoom = NULL,
  scale = FALSE
  )
```

Arguments

input.data	a dataframe with one time series per column.
response	character string, name of the numeric column to be used as response in the model.
drivers	character vector, names of the numeric columns to be used as predictors in the model.
time	character vector, name of the numeric column with the time/age.
oldest.sample	character string, either "first" or "last". When "first", the first row taken as the oldest case of the time series and the last row is taken as the newest case, so ecological memory flows from the first to the last row of input.data. When "last", the last row is taken as the oldest sample, and this is the mode that should be used when input.data represents a palaeoecological dataset. Default behavior is "first".
lags	numeric vector of positive integers, lags to be used in the equation. Generally, a regular sequence of numbers, in the same units as time. The use seq to define it is highly recommended. If 0 is absent from lags, it is added automatically to allow the consideration of a concurrent effect. Lags should take into account the temporal resolution of the data, and be aligned to it. For example, if the interval between consecutive samples is 100 years, lags should be something like 0, 100, 200, 300. Lags can also be multiples of the time resolution, such as 0, 200, 400, 600 (in the case time resolution is 100 years).
time.zoom	numeric vector of two numbers of the time column used to subset the data if desired.

scale	boolean, if TRUE, applies the scale function to normalize the data. Required if the lagged data is going to be used to fit linear models.

Details

The function interprets the time column as an index representing the

Value

A dataframe with columns representing time-delayed values of the drivers and the response. Column names have the lag number as a suffix. The response variable is identified in the output as "Response_0".

Author(s)

Blas M. Benito <blasbenito@gmail.com>

See Also

computeMemory

Examples

```
#loading data
data(palaeodata)

#adding lags
lagged.data <- prepareLaggedData(
 input.data = palaeodata,
 response = "pollen.pinus",
 drivers = c("climate.temperatureAverage", "climate.rainfallAverage"),
 time = "age",
 oldest.sample = "last",
 lags = seq(0.2, 1, by=0.2),
 time.zoom=NULL,
 scale=FALSE
)
str(lagged.data)
```

---

runExperiment *Computes ecological memory patterns on simulated pollen curves produced by the* virtualPollen *library.*

Description

Applies computeMemory to assess ecological memory on a large set of virtual pollen curves.

Usage

```
runExperiment(
 simulations.file = NULL,
 selected.rows = 1,
 selected.columns = 1,
 parameters.file = NULL,
 parameters.names = NULL,
 sampling.names = NULL,
 driver.column = NULL,
 response.column = "Response_0",
 subset.response = "none",
 time.column = "Time",
 time.zoom = NULL,
 lags = NULL,
 repetitions = 10
 )
```

Arguments

simulations.file	list of dataframes, output of the function simulatePopulation of the virtualPollen library.
selected.rows	numeric vector, rows (virtual taxa) of simulations.file to be analyzed.
selected.columns	numeric.vector, columns (experiment treatments) of simulations.file to be analyzed.
parameters.file	dataframe of simulation parameters.
parameters.names	vector of character strings with names of traits and niche features from parameters.file to be included in the analysis (i.e. c("maximum.age", "fecundity", "niche.A.mean", "niche.A.sd"))
sampling.names	vector of character strings with the names of the columns of simulations.file.
driver.column	vector of character strings, names of the columns to be considered as drivers (generally, one of "Suitability", "Driver.A", "Driver.B).
response.column	character string defining the response variable, typically "Response_0".
subset.response	character string, one of "up", "down" or "none", triggers the subsetting of the input dataset. "up" only models ecological memory on cases where the response's trend is positive, "down" selectes cases with negative trends, and "none" selects all cases.
time.column	character string, name of the time/age column. Usually, "Time".
time.zoom	numeric vector with two numbers defining the time/age extremes of the time interval of interest.

lags	ags numeric vector of positive integers, lags to be used in the equation. Generally, a regular sequence of numbers, in the same units as time. The use seq to define it is highly recommended. If 0 is absent from lags, it is added automatically to allow the consideration of a concurrent effect. Lags should take into account the temporal resolution of the data, and be aligned to it. For example, if the interval between consecutive samples is 100 years, lags should be something like 0, 100, 200, 300. Lags can also be multiples of the time resolution, such as 0, 200, 400, 600 (in the case time resolution is 100 years).
repetitions	integer, number of random forest models to fit.

Value

A list with 2 slots:

- names matrix of character strings, with as many rows and columns as simulations.file. Each cell holds a simulation name to be used afterwards, when plotting the results of the ecological memory analysis.
- output a list with as many columns and columns as simulations.file. Each slot holds a an output of computeMemory.
  - memory dataframe with five columns:
    - Variable character, names and lags of the different variables used to model ecological memory.
    - median numeric, median importance across repetitions of the given Variable according to Random Forest.
    - sd numeric, standard deviation of the importance values of the given Variable across repetitions.
    - min and max numeric, percentiles 0.05 and 0.95 of importance values of the given Variable across repetitions.
  - R2 vector, values of pseudo R-squared value obtained for the Random Forest model fitted on each repetition. Pseudo R-squared is the Pearson correlation beteween the observed and predicted data.
  - prediction dataframe, with the same columns as the dataframe in the slot memory, with the median and confidence intervals of the predictions of all random forest models fitted.
  - multicollinearity multicollinearity analysis on the input data performed with vif. A vif value higher than 5 indicates that the given variable is highly correlated with other variables.

Author(s)

Blas M. Benito <blasbenito@gmail.com>

See Also

computeMemory

Index

∗ **datasets**
climate, 2
palaeodata, 10
palaeodataLagged, 11
palaeodataMemory, 12
pollen, 16

climate, 2, *10*
computeMemory, 3, *5–8*, *11, 12*, *15, 16*, *18*, *20*

experimentToTable, 5, *7, 8*
extractMemoryFeatures, *5*, 6

loess, *9*

mergePalaeoData, 9, *10*

palaeodata, 10, *11*
palaeodataLagged, *3*, 11, *12*
palaeodataMemory, 12
plotExperiment, *6*, 13
plotInteraction, 14
plotMemory, *5*, *14*, 15
pollen, *10*, 16
prepareLaggedData, *3*, *7*, *11*, 17

randomForest, *14, 15*
ranger, *4*, *14, 15*
rnorm, *4*
rpart, *14, 15*
runExperiment, *5, 6*, *13, 14*, 18

scale, *18*
seq, *17*, *20*

vif, *4*, *12*, *20*